Linearity-Preserving Limiters on Irregular Grids

Marsha Berger
Courant Institute
New York, NY 10012
berger@cims.nyu.edu

Michael Aftosmis
NASA Ames
Moffett Field, CA 94035
aftosmis@nas.nasa.gov

Scott Murman
ELORET/NASA Ames
Moffett Field, CA 94035
smurman@nas.nasa.gov

Abstract

This paper examines the behavior of flux and slope limiters on non-uniform grids in multiple dimensions. We note that on non-uniform grids the scalar formulation in standard use today sacrifices k-exactness, even for linear solutions, impacting both accuracy and convergence. We rewrite some well-known limiters in a n way to highlight their underlying symmetry, and use this to examine both traditional and novel limiter formulations. A consistent method of handling stretched meshes is developed, as is a new directional formulation in multiple dimensions for irregular grids. Results are presented demonstrating improved accuracy and convergence using a combination of model problems and complex three-dimensional examples.

1 Introduction

For many finite-difference or finite-volume computations with discontinuous solutions limiters are a necessary evil. On one hand they suppress oscillations and maintain monotone solutions. On the other hand, they reduce accuracy and can hamper convergence to steady state [2]. Limiters can be difficult to extend to multidimensional unstructured grids, and they do not degenerate gracefully on distorted grids. Other problems can include limiter chatter, rattling in near-constant portions of the flow field, and additional convergence problems with Newton's method for non-smooth limiters [13]. These problems can occur on both structured and unstructured grids, for both flux and slope-limiter formulations.

Most of the research on limiters has been in one space dimension, where a reliable theory exists [6]. There is substantially less literature on extensions to two or more dimensions, where theory is lacking. On multi-dimensional structured grids, limiters are typically applied independently in each coordinate direction, which works well in practice. On unstructured grids most codes use a scalar limiter, where the gradient is reduced by the same scalar fraction regardless of direction [3]. There is almost no discussion on the application of limiters to irregular or non-uniform grids, although nearly all meshes on real geometries fall into this category (some recent exceptions however are [7, 9]). The purpose of this paper is to develop limiters that preserve monotonicity, are exact for linear solutions (called k-exactness for k=1), and have good convergence properties on non-smooth grids. We illustrate their performance with examples computed on multi-level Cartesian grids with embedded boundaries. These grids make a good testbed, since they include large portions of regular uniform grid cells, mesh interfaces with 2:1 mesh stretching, as well as general irregular polyhedra at the embedded boundaries.

In the rest of this section we briefly summarize some relevant parts of the literature. Section 2 presents some problems with limiters in common use today, which may not be well-known. Section 3 discusses a solution to the problem of limiters on stretched meshes. Generalizations of one dimensional limiters for multi-dimensional irregular grids are presented in section 4. Computational results using both model problems and a variety of 3D test cases are presented in section 5.

1.1 Background

The most common framework for looking at limiters was systematized by Sweby [12], using the flux formulation. By analogy with Flux Corrected Transport [14], one writes a second-order scheme as a first-order scheme plus a limited anti-diffusive flux, shown here for the case of linear advection $u_t + au_x = 0$, as

$$u_i^{n+1} = u_i^n - \lambda(u_i - u_{i-1}) - (\psi_i F_{i+1/2} - \psi_{i-1} F_{i-1/2})$$

where the flux $F_{i+1/2} = \frac{1}{2}\lambda(1-\lambda)(u_{i+1} - u_i)$. Here the ψ_i's are the flux limiters, and $\lambda = \frac{a\Delta t}{\Delta x}$.

Define the ratio R of forward to backward differences in the solution (note that R is the inverse of Sweby's r) as

$$R_i = \frac{u_{i+1} - u_i}{u_i - u_{i-1}}. \tag{1}$$

By choosing $\psi_i = \psi(R_i)$ properly, a total-variation-diminishing (TVD) scheme can be achieved. Sweby then derives the well-known result

$$0 \le (\frac{\psi(R)}{R}, \psi(R)) \le 2.$$

This is often expressed graphically for the different possible limiters using the *Sweby diagram* of figure 1(a): choosing ψ in the shaded region leads to a second-order TVD solution. An important point to note is that for second order accuracy away from extrema, the limiter must satisfy

$$\psi(1) = 1.$$

In other words, the method should not do any limiting when the solution is a linear function. Also ψ must be symmetric,

$$\frac{\psi(R)}{R} = \psi(\frac{1}{R}), \tag{2}$$

although this symmetry may not be immediately apparent in figure 1(b). Later we will rewrite the limiters in a more transparently symmetric form. All the limiters below satisfy

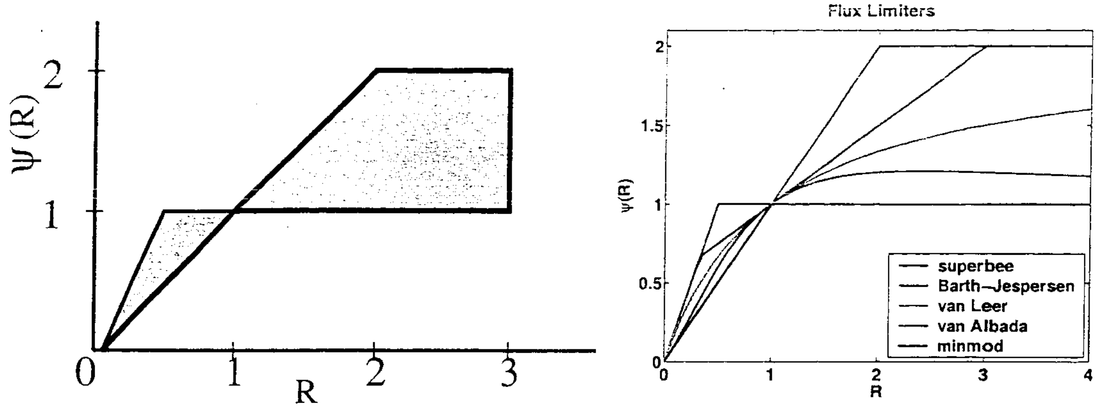


Figure 1: (a) Second order TVD region from [12]. (b) Common flux limiters ψ from eq. (5).

the symmetry relation (2). Limiters of interest, in order of increasing dissipation, are

$$\text{(superbee)} \qquad \psi_{sb}(R) = \text{minmod}[\max(1, R), 2, 2R] \tag{3}$$

$$\text{(Barth} - \text{Jespersen)} \qquad \psi_{BJ}(R) = \frac{1}{2}(R+1)\min[\min(1, \frac{4R}{R+1}), \min(1, \frac{4}{R+1})] \tag{4}$$

$$\text{(van Leer)} \qquad \psi_{VL}(R) = \frac{|R|+R}{|R|+1} \tag{5}$$

$$\text{(van Albada)} \qquad \psi_{VA}(R) = \frac{R^2+R}{R^2+1} \tag{6}$$

$$\text{(minmod)} \qquad \psi_{mm}(R) = \max\{0, \min(1, R)\} \tag{7}$$

The superbee limiter follows the boundary of the TVD region, and is included in figure 1(b) for reference. The superbee and Barth-Jespersen limiters are the most compressive, and are known to turn smooth waves into square waves. In multiple dimensions their overly compressive nature may lead to staircasing of discontinuities on non-aligned grids.

Spekreijse [11] shows the equivalence of flux limiting with slope limiting, which is more natural for finite volume schemes in *REP* form [5]: *Reconstruct* the solution from its cell averages, *Evolve* the reconstructed solution from time t_n to t_{n+1}, and *Project* the solution back onto the grid to update the integral cell averages at the new time. We can see this by computing states in the flux form for linear advection, using the one-sided differences from the Fromm scheme. Computing the states at the cell interfaces we get

$$u^L_{i+1/2} = u_i + \frac{1}{2}\psi(R_i)\,(u_i - u_{i-1}) \tag{8}$$

$$u^R_{i-1/2} = u_i - \frac{1}{2}\psi(1/R_i)\,(u_{i+1} - u_i) \tag{9}$$

where $u^L_{i+1/2}$ is the left state at the right edge, etc. By requiring that this be equivalent to limiting a single reconstructed gradient at cell i, here the central difference $(u_{i+1} - u_{i-1})/2h$,

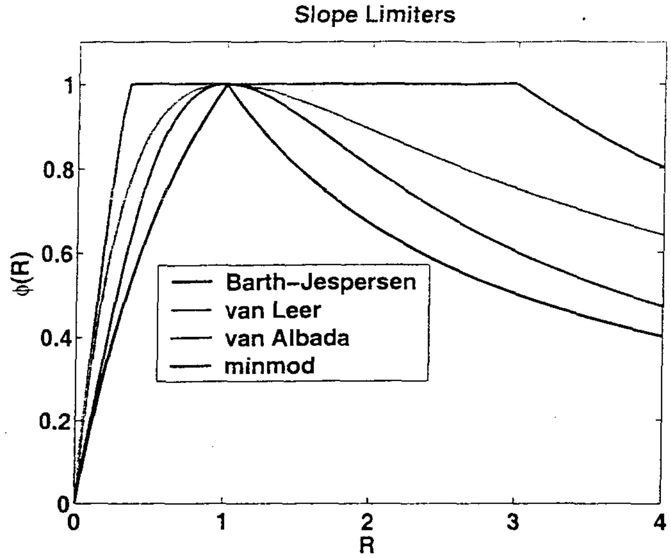


Figure 2: Common limiters in slope form from eq. (14).

we get

$$\begin{aligned} u^L_{i+1/2} &= u_i + \frac{1}{2}\phi(R_i)\ (u_{i+1} - u_{i-1})/2 \\ u^R_{i-1/2} &= u_i - \frac{1}{2}\phi(1/R_i)(u_{i+1} - u_{i-1})/2 \end{aligned} \quad (10)$$

Thus along with using eq. (2), we get the relationship between a flux limiter ψ and a slope limiter ϕ

$$\psi(R) = \phi(R)(\frac{R+1}{2}). \quad (11)$$

Whereas flux limiters satisfy $0 \leq \psi(R) \leq 2$, a slope limiter satisfies $0 \leq \phi(R) \leq 1$. Slope limiters make the reconstruction step easy, by writing the reconstructed function $U_i(x)$ in cell i as $U_i(x) = u_i + \phi(R_i)\nabla u_i(x - x_i)$, where u_i is the cell average. Note that for any limiter ψ satisfying eq. (2), ϕ satisfies the slope form of the symmetry condition

$$\phi(R) = \phi(\frac{1}{R}). \quad (12)$$

Figure 2 shows the behavior of the same collection of limiters as figure 1 but this time in their slope form. The common limiters in slope form are:

$$\begin{aligned} \phi_{BJ}(R) &= \max\{0, \min(2R, 1), \min(R, 2)\} & (13) \\ \phi_{VL}(R) &= \frac{4R}{(R+1)^2} & (14) \\ \phi_{VA}(R) &= \frac{2R}{R^2+1} & (15) \\ \phi_{mm}(R) &= \max\{0, \min(\frac{2R}{1+R}, \frac{2}{1+R})\} & (16) \end{aligned}$$

1.2 Symmetric View of Limiters

While slope limiters of the form in figure 2 are helpful, we briefly digress to present a more intuitive way to graph the slope limiters. In figure 3 the independent variable f is the fraction $u_i - u_{i-1}$ of the whole jump $u_{i+1} - u_{i-1}$. This form has the added benefit that we can more easily see the behavior of the limiter for large R.

Let the undivided differences $\triangle_- = u_i - u_{i-1}$, and $\triangle_+ = u_{i+1} - u_i$, so that $\triangle_c = u_{i+1} - u_{i-1} = \triangle_+ + \triangle_-$. The fraction $f = \triangle_-/\triangle_c$ gives a symmetric view of the slope limiters as f ranges from 0 to 1. The symmetry is more apparent since the limiter can be plotted as a function of $\triangle_-$ or $\triangle_+$, and the function is symmetric about $f = 1/2$. It also becomes obvious that it is the neighbor of u_i with the larger jump, either u_{i+1} or u_{i-1} that causes the limiting of u_i's slope, since it is *this* jump that would cause the overshoot at the *other* neighbor's face. In these variables the common limiters become

$$\begin{aligned}
\phi_{VL} &= \frac{\triangle_+ u_i \, \triangle_- \, u_i}{\triangle_c^2 u_i} \\
\phi_{sin} &= \sin(\frac{\pi \triangle_-}{\triangle_c}) \\
\phi_{VA} &= \frac{2\,\triangle_+ \, \triangle_-}{\triangle_+^2 + \triangle_-^2} \\
\phi_{minmod} &= \min\{\frac{\triangle_+}{\triangle_c}, \frac{\triangle_-}{\triangle_c}\}
\end{aligned} \qquad (17)$$

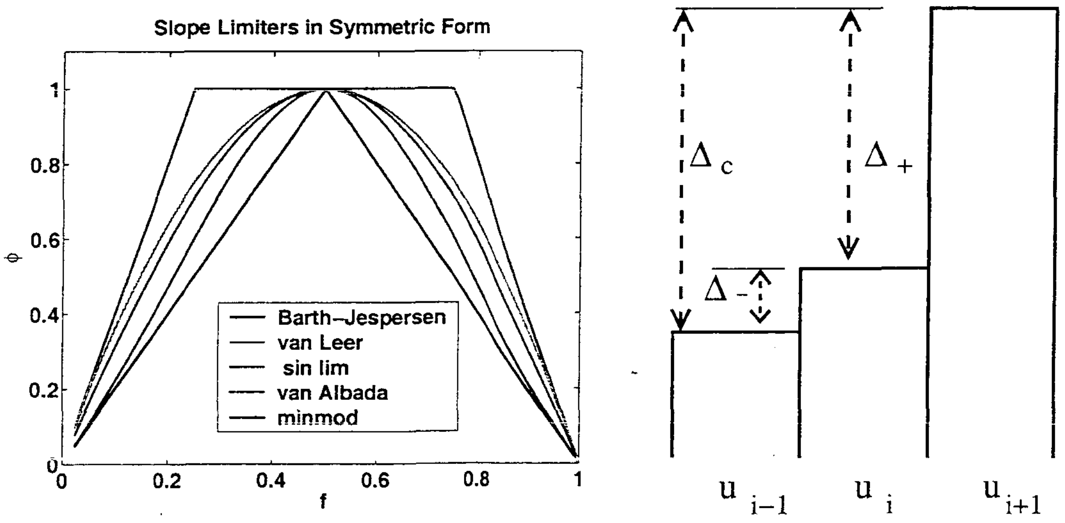


Figure 3: (a) Symmetric form of slope limiters. (b) Cell averages corresponding to the limiters. When u_i is exactly in the middle, the data is linear, $f = .5$, and the limiter is 1.

This symmetric form also makes it easy to devise new limiters. For example, included in figure 3(a) is a new *sin* limiter, defined by

$$\begin{aligned}\phi_{sin}(R) &= \sin(\frac{\pi\triangle_-}{\triangle_c}) = \sin(\frac{\pi(\triangle_c - \triangle_+)}{\triangle_c}) = \cos(\pi)\sin(\frac{-\pi\,\triangle_+}{\triangle_c}) \\ &= \sin(\frac{\pi\triangle_+}{\triangle_c})\end{aligned}$$

This limiter is continuously differentiable, does not limit linear data, satisfies the symmetry property (12), and falls between the van Leer and van Albada limiters in terms of dissipation.

1.3 Multi-dimensional Formulations

On structured grids in two space dimensions Spekreijse limits each coordinate direction separately and the scheme preserves monotonicity at steady state. However Saltzman [10] shows that for the second-order time-accurate MUSCL scheme in two dimensions, a larger neighborhood of values, illustrated in two dimensions in figure 4, needs to be included in the limiter for strict monotonicity of the time-dependent solution. He defines the new *Benjamin* limiter by extending Barth-Jespersen (also called the limited monotonicized central differences) to include corner coupling. A one-dimensional geometric interpretation of Barth-Jespersen is that the central difference cannot exceed twice the forward or backward differences at that cell. (From this it follows that the reconstructed value at the face does not exceed the neighbor's cell-centered value). In two dimensions, the central difference must also be checked against the forward and backward differences of neighboring rows and columns to ensure there are no new maxima at the next time step. In Saltzman's notation, but simplified here by assuming all gradients are positive, the limited gradient in the x direction can be written

$$\begin{aligned}h\,\phi_i\,u_{ij_x} = \min\{ \quad & 2(u_{i+1,j} - u_{i,j}),\ 2(u_{i+1,j+1} - u_{i,j+1}),\ 2(u_{i+1,j-1} - u_{i,j-1}), \\ & 2(u_{i,j} - u_{i-1,j}),\ 2(u_{i,j+1} - u_{i-1,j+1}),\ 2(u_{i,j-1} - u_{i-1,j-1}), \\ & 0.5(u_{i+1,j} - u_{i-1,j})\} \qquad (18)\end{aligned}$$

Similarly, the limited y gradient is either the central difference in the j direction, or the minimum of twice the forward or backward difference in any neighboring column in that direction. This limiter has been found to be overly diffusive for many applications, but we will use elements reminiscent of it later in devising a limiter for cut cells.

Although limiters can be too dissipative they can also be too compressive, a well-known problem for example with the superbee or Barth-Jespersen limiters. In PPM, the higher order reconstruction scheme of [4], this was handled by explicitly adding second difference dissipation at strong shocks in both the normal and the tangential direction. However one of our goals in section 4 will be to devise a more dissipative limiter to automatically accomplish this on irregular grids.

2 Problems with Limiters

Despite the simple framework sketched in the previous section, the fact remains that the most common situations found in practice are more complex. Little work appears in the literature on applying limiters to irregular meshes in one or more dimensions. This includes

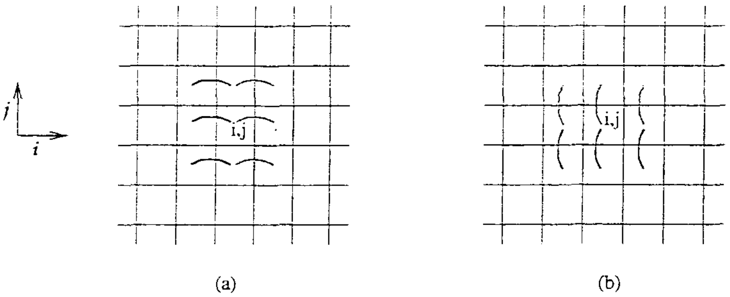


Figure 4: Stencil for the Benjamin limiter of eq. (18) for two-dimensional time-dependent problems. The gradient computed at i, j is compared with the forward differences (in green) and the backward differences (in purple) for the limiting in the x direction (a), and similarly for the y direction (b).

their application on unstructured meshes, although these grids are now nearly as common as their structured counterparts. In this section we illustrate some problems with limiters that are either not well known or ignored since practical solutions are not available.

2.1 Stretched Meshes

When the grid is unstructured or irregular, a common procedure to approximate the gradient at cell i is to use a least squares fit to the solution in neighboring cells [3]. In general this gives a first-order accurate approximation, and for linear data it is exact, even on distorted meshes. However the limiting procedure can destroy this property.

We illustrate this in one-dimension using three unequally spaced adjacent grid cells. The least squares gradient approximation at cell i is

$$\nabla u_i = \frac{h_+^2}{h_+^2 + h_-^2} D_+ u_i + \frac{h_-^2}{h_+^2 + h_-^2} D_- u_i \tag{19}$$

where $h_+ = x_{i+1} - x_i$, $h_- = x_i - x_{i-1}$, $D_+ u_i = (u_{i+1} - u_i)/h_+$, etc. Note that the least squares gradient in eqn. (19) is a convex combination of the forward and backward differences computed on the irregular mesh. If the mesh is uniform, $h_+ = h_-$, the weights become 1/2 each, and we recover the familiar second-order first derivative stencil

$$D_c u_i = \frac{(D_+ u_i + D_- u_i)}{2} = \frac{u_{i+1} - u_{i-1}}{2h} \tag{20}$$

On a non-uniform mesh a second order accurate gradient approximation is of course possible, but it does not minimize the least squares error of the linear fit. The second order non-uniform approximation is a linear combination with different weights, $\nabla u_i = \frac{h_+}{h_+ + h_-} D_+ u_i + \frac{h_-}{h_+ + h_-} D_- u_i$.

Consider the one dimensional stretched mesh of figure 5, where the cell size of u_i is h. In the left figure the mesh is locally irregular, and the cell size of u_{i+1} is h/2 In the figure on the right, the mesh has a constant stretching ratio of 2. If eq. (1) is applied to linear data with slope s,

$$R = \frac{u_{i+1} - u_i}{u_i - u_{i-1}} = \frac{s \cdot (x_{i+1} - x_i)}{-s \cdot (x_i - x_{i-1})} = \frac{3/4h}{h} = \frac{3}{4}. \tag{21}$$

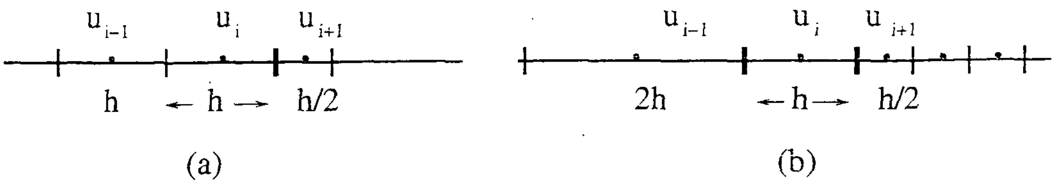


Figure 5: The left mesh has a mesh refinement interface; the mesh on the right has a constant stretching ratio. The ratio of the difference $R = \frac{u_{i+1}-u_i}{u_i-u_{i-1}}$ is 3/4 in (a) or 1/2 in (b). For linear data a ratio of 1.0 is expected by the uncorrected slope or flux limiter forms in eq. (14) or (5).

where the cell average value u_i is taken to be located at the cell center x_i. On a uniform grid this ratio would be 1 but on a stretched mesh it isn't, so most of the limiters (except Barth-Jespersen or superbee) will be activated. For example, the van Leer limiter of (14) gives a limiter value of 6/7 when $R = 3/4$. For the grid in figure 5(b) R=1/2, and the limiter value will be 2/3. The Barth-Jespersen limiter, which has the geometric interpretation that the reconstruction at the cell edge should not exceed the cell centered value of the adjacent cell does not require limiting on stretched meshes for any stretching ratio.

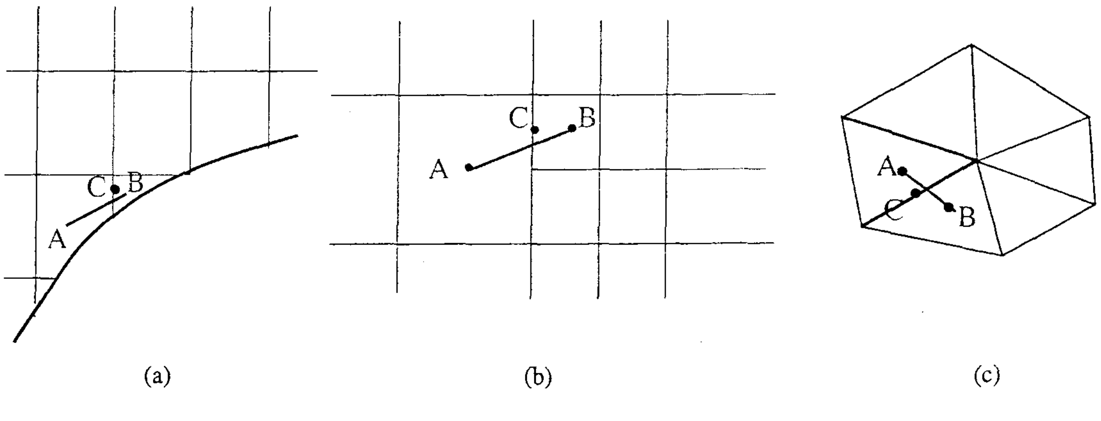


Figure 6: In all three examples, linear data is not preserved by standard limiters. Let the line between cell centers A and B also be the level lines of the solution. The flux evaluation and solution limiting occurs at point C, which is not on the line. Point C will have an overshoot, and the *exact* gradient which was reconstructed at A and B will be limited.

This limiting of linear solutions occurs often in higher dimensions, and is responsible for some loss of accuracy. The limiting is occurring not because of extrema or discontinuities, but simply because of the mesh stretching. Figure 6 shows three situations where this can occur. In each case, let the linear data be such that points A and B are equal. Point C will then be an extremum when compared to the cell averages, and the gradients at both A and

B will be limited. This happens regardless of which limiter is used, since all limiters will see C as a new extremum. On smoothly varying triangular grids where the median dual is approximated with the centroidal dual, the midpoint of the edge is only $O(h^2)$ from the point on the line between the two neighbors, so this is not likely to be noticeable. On the other hand, the use of scalar limiters in multiple dimensions will exacerbate this problem.

2.2 Face-based Limiters

One idea that has been previously suggested to reduce the dissipation from limiters is to apply them on a face-by-face basis. If one neighbor causes an adjacent cell to limit its gradient, the limiter would only be applied on its common face - that is, on one side of the reconstruction. The neighbor on the other side might allow some or all of the gradient to be used on the face it shares with the same cell. This can be thought of as a generalization of the geometric interpretation of the Barth-Jespersen limiter. Note that this limiting procedure does not preserve the mean of the reconstruction. Face-based limiting was shown not to work well in practice in [2]. Here we explain this by showing that this approach does not preserve monotonicity.

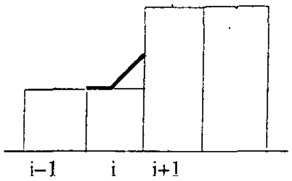


Figure 7: Initial piecewise constant solution for counter-example showing that face-based limiting does not preserve monotonicity. All gradients are zero except at cell i.

Let u be a step function as shown in figure 7. For linear advection, $u_t + u_x = 0$, the update at cell i using a spatially second order upwind differencing (so the upwind state is on the left) and face-based limiting gives

$$\begin{aligned} u_i^{n+1} &= u_i^n - \frac{\Delta t}{\Delta x}(u^L_{i+1/2} - u^L_{i-1/2}) \\ &= u_i^n - \frac{\Delta t}{\Delta x}(u^L_{i+1/2} - u^n_{i-1}) \\ &= u_i^n - \lambda((u_i^n + h/2\,\phi_R\, u_{i_x}) - u^n_{i-1}), \end{aligned} \tag{22}$$

where ϕ_R is the value of the limiter on the right face of cell i. Let u_{i_x} be the central difference for cell i, $u_{i_x} = \frac{u_{i+1}-u_{i-1}}{2h}$. Note that the forward difference $D_+ u_i = \frac{u_{i+1}-u_i}{h} < u_{i_x}$ is larger than this, so the reconstruction at the right edge will not exceed the value u_{i+1} and will not need to be limited using Barth-Jespersen. Hence $\phi_R = 1$, so

$$\begin{aligned} u_i^{n+1} &= u_i^n - \lambda(u_i + h/2\frac{(u_{i+1} - u_{i-1})}{2h} - u^n_{i-1}) \\ &= u_i^n - \lambda(u_i - u_{i+1}/4 + u_{i-1}/4 - u_{i-1}) \\ &= u_i^n - \lambda(1/4\, u_{i+1} + u_i - 5/4\, u_{i-1}) \\ &= u_i^n - \lambda(u_{i+1}/4 - u_{i-1}/4) \quad \text{since} \quad u_{i-1} = u_i \\ &< u_i^n \quad \text{since} \quad u_i < u_{i+1} \end{aligned} \tag{23}$$

In other words, there will be an undershoot at u_i at the next time step. As this example shows, the face with the largest change in the solution is responsible for limiting at the other face. This does not occur with face-based limiting, but occurs naturally for symmetric limiters satisfying eq. (12).

2.3 Directional Limiting and Gradient Rotation

Most finite volume implementations on unstructured grids use a scalar form of the limiter

$$u^L = u_i + \phi_i \overrightarrow{r} \nabla u_i \tag{24}$$

here ϕ is a scalar slope limiter in cell i and $\overrightarrow{r}$ is the position vector from the cell centroid to the face. This extension of (10) to multiple dimensions is in contrast to the traditional structured grid implementations which limit the gradients independently in each coordinate direction. Thus, if the data is linear in one coordinate direction, the gradient will not be reduced regardless of limiter activity in the other directions. By contrast, the scalar limiter, while popular, is far more severe since limiting triggered by *any* face of a polyhedral cell degrades the gradient in *all* directions. Results in [2] show that even for smooth flows, the scalar implementation is dramatically more dissipative than the directional form.

While directional limiting is straightforward on cells with orthogonal faces, it turns out to be somewhat more subtle on more general meshes. The underlying difficulty stems from competing requirements of a face-by-face implementation that still guarantees positivity. Figure 8 illustrates the situation for both cell types. The frames on the left of figure 8a) trace the evolution of the gradient on a Cartesian cell, while the frames on the right examine the process for a cell with non-orthogonal faces (in this case simply a triangle).

Following the evolution on the orthogonal cell in 8(a), frame 8(a.1) shows the monotonicity boundary imposed by *face1*. The directional implementation removes the component of ∇u normal to this face (shown in red) resulting in the modified gradient $\nabla u'$. In frame 8(a.2), the gradient is further reduced by removal of the component of $\nabla u'$ that extends beyond the monotonicity boundary established by *face2*. In reviewing this process, notice that since faces 1 and 2 are orthogonal, the limiting in frame 8.a.2 retards the gradient while still respecting the limit boundary from *face1*. The limited gradient, $\overrightarrow{\phi} \nabla u$ resulting from this face-based implementation sits on the perimeter of the monotonicity region (shaded) in figure 8(a.2).

By contrast, examine the result of this process on the general polyhedral cell on the right of figure 8. In frame 8(b.1) *face1* limits ∇u to $\nabla u'$ by again removing the component normal to the monotonicity boundary. In frame 8(b.2) we impose the corresponding limit from *face2*. This time, however, removal of the face normal component of the gradient produces a limited gradient $\overrightarrow{\phi} \nabla u$ which violates the earlier boundary established by *face1*, and the final limited gradient unfortunately lies *outside* the monotonicity region for the cell.

The reduction in dissipation and improved reconstruction properties offered by directional limiting [2] gives strong motivation to successfully implement them on general polygonal meshes. However, as the sketches in figure 8 demonstrate, this implementation is delicate. In essence, each new face must respect the boundary of the monotonicity region established by all other faces. This poses an implicit problem for the limiter values. For each cell gradient, we seek the point on the boundary of the monotonicity region which preserves as much of the unlimited gradient as possible. This problem can be formulated

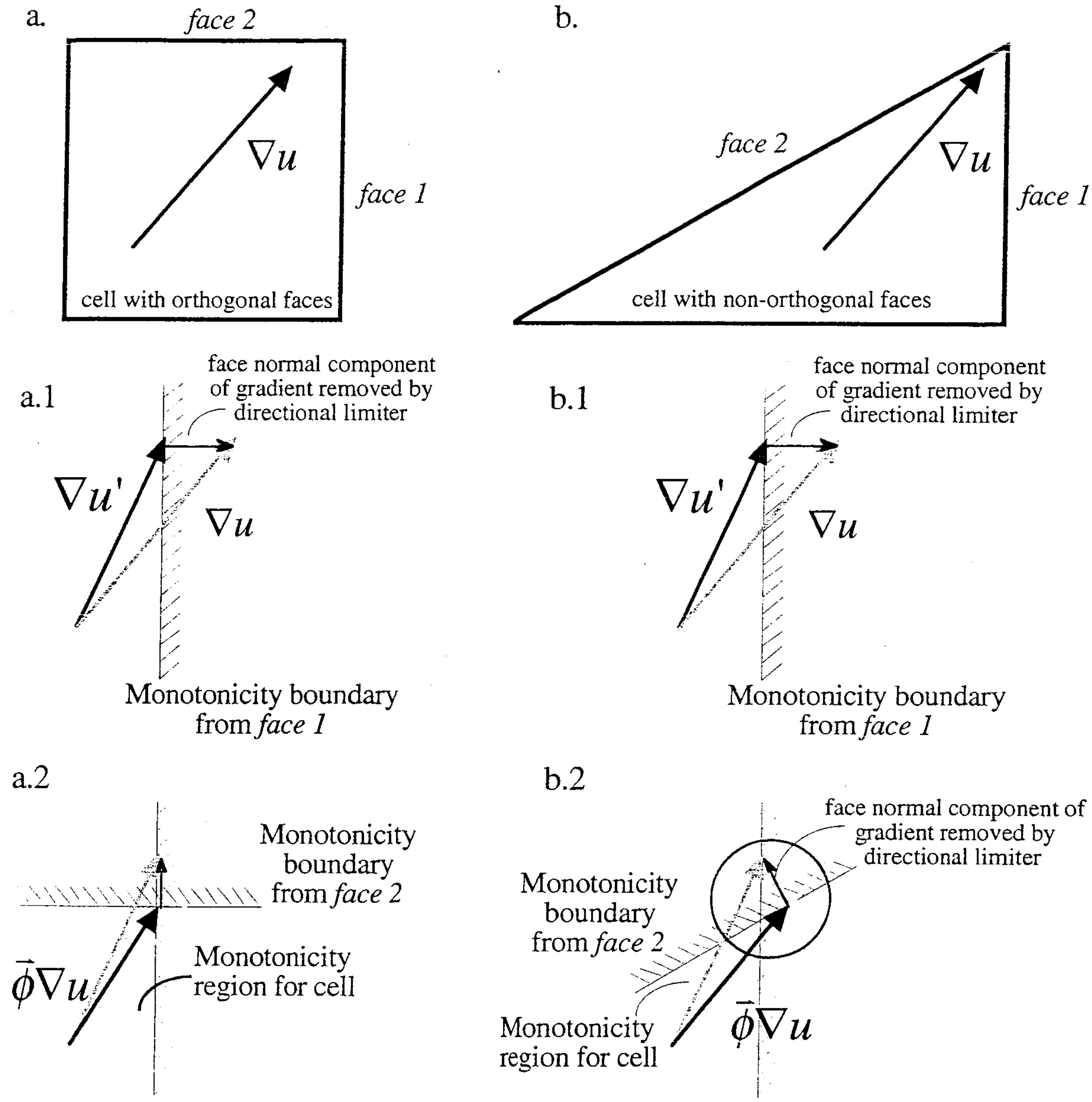

a.
face 2
∇u
face 1
cell with orthogonal faces
b.
face 2
∇u
face 1
cell with non-orthogonal faces
a.1
face normal component of gradient removed by directional limiter
∇u'
∇u
Monotonicity boundary from face 1
b.1
face normal component of gradient removed by directional limiter
∇u'
∇u
Monotonicity boundary from face 1
a.2
Monotonicity boundary from face 2
$\vec{\phi}\nabla u$
Monotonicity region for cell
b.2
face normal component of gradient removed by directional limiter
Monotonicity boundary from face 2
Monotonicity region for cell
$\vec{\phi}\nabla u$

Figure 8: Illustration of the limiting process on a cell with orthogonal (a) and non-orthogonal (b) faces. After limiting against two faces, the resulting region shaded in yellow shows the allowable monotonicity region for the gradient. The limited gradient must fall within in the yellow region.

as a clipping operation, or as an implicit system of equations. Either of these cases requires an additional sweep over the cell faces to determine the limiter value which simultaneously respects the monotonicity boundaries imposed by all faces of general polyhedral cells. On cells with perpendicular faces, this system decouples and orthogonality ensures that limiting of one face will not result in an overshoot of a boundary that has already been imposed.

3 Limiters on Stretched Meshes

It is straightforward to rewrite the limiters to preserve linear solutions on stretched meshes. We use scaled differences, or gradients, that incorporate the mesh stretching in the limiter definitions instead of using undivided differences. In other words, replace $\triangle_+ \rightarrow D_+, \triangle_- \rightarrow D_-$ and $\triangle_c \rightarrow .5D_c$. For the van Leer limiter for an increasing solution ($|R| = R > 0$) on a uniform mesh for example, we get

$$\phi_{VL}(R) \;=\; \frac{4R}{(R+1)^2} = \frac{4(u_{i+1}-u_i)(u_i-u_{i-1})}{(u_{i+1}-u_{i-1})^2} = \frac{4\,\triangle_+\,\triangle_-}{\triangle_c^2} = \frac{D_+D_-}{D_c^2} \tag{25}$$

Using this form on a stretched mesh with linear data will preserve the linearly-exact gradient reconstruction, since D_+, D_- and D_c will all be equal to the exact slope, and no limiting will take place.

Re-writing of the common limiters to reflect the mesh stretching gives

$$\phi_{VL} \;=\; \frac{D_+u_iD_-u_i}{D_c^2u_i} \tag{26}$$

$$\phi_{sin} \;=\; \sin(\frac{\pi D_-}{D_c}) \tag{27}$$

$$\phi_{VA} \;=\; \frac{2D_+D_-}{D_+^2+D_-^2} \tag{28}$$

$$\phi_{minmod} \;=\; \min\{\frac{D_+}{D_c},\frac{D_-}{D_c}\} \tag{29}$$

On a regular mesh, where $D_+ + D_- = 2\,D_c$, eq. (25) is of the form

$$\phi = ab/(a+b)^2, \tag{30}$$

which is easily seen to give $\phi \leq 1$ for all a, b. On a stretched mesh the central difference is replaced by the least squares difference (however we still use the notation D_c), so D_c is not the sum of the forward and backward differences, and eq. (25) does not lie between 0 and 1. One more step is needed to turn this into a limiter. The function is brought back into the form (30) by writing

$$\phi_{VL} = \min\{\frac{D_+u_i\cdot(D_c-D_+)u_i}{D_c^2u_i},\frac{D_-u_i\cdot(D_c-D_-)u_i}{D_c^2u_i}\}. \tag{31}$$

Eq. (31) computes an estimate of the limiter from both sides of the cell, and takes the minimum. A geometric interpretation of this is that each face computes the limiters for its adjacent cells assuming the cell it *doesn't* see (on the other side of the other face) is the same size as it is. The *min* is needed in case this isn't true – one face will compute a more stringent limiter value than the other side. In fact, the side with the greater change in the solution will cause the limiter to fire at the other side of the cell.

Eq. (31) is more complicated than simply defining $R' = D_+/D_-$ and using the same limiter definitions. That is because in an edge-based unstructured code, this is actually the way the limiter is implemented, since information from two cells away is not readily available. Alternatively one can try to adjust the weights in the least squares formulation so that even on a stretched mesh eq. (20) holds. However, this looked unlikely to extend to more than one dimension and was not pursued further.

4 Multidimensional Limiting

Several approaches to non-directionally aligned limiting in multiple dimensions have been proposed. Recently, some researchers have used a weighted least squares to limit the gradient, or even construct it initially. In [8] data-dependent least squares is considered, with weights chosen geometrically based on the distance between neighboring cells. Rider and Kothe [9] have suggested incorporating monotonicity constraints into the least squares gradient procedure itself. Their algorithm first computes the least squares gradient, then makes a second pass where only neighboring data with directional derivative smaller than the least squares gradient in that direction is used. This can be considered a generalization of the *min* limiter. They show that a very close approximation to this is obtained by solving a weighted least squares problem where the weights are inversely proportional to the change in the solution $u_k - u_0$ at each data point. Our very limited experience with this seems to indicate it works well but is prohibitively expensive, since a new least squares problem has to be solved for every state variable in every direction at every iteration.

In these next subsections we present two simpler, and rather different approaches to multi-dimensional limiting. They have different implementation costs and benefits, so we present them both (and use them both in the flow solver in the examples of Section 5).

4.1 Aligning the Data

When the data is not coordinate aligned, one strategy is to make it so by creating a new *virtual* cell center. This virtual point should be aligned along the single coordinate direction in which the limiting should be performed. Essentially, we *re-center* the data to place it on the normal through the face centroid where the flux evaluation is performed. Note however that the re-centering procedure itself uses a different component of the gradient. This becomes one large implicit system of equations.

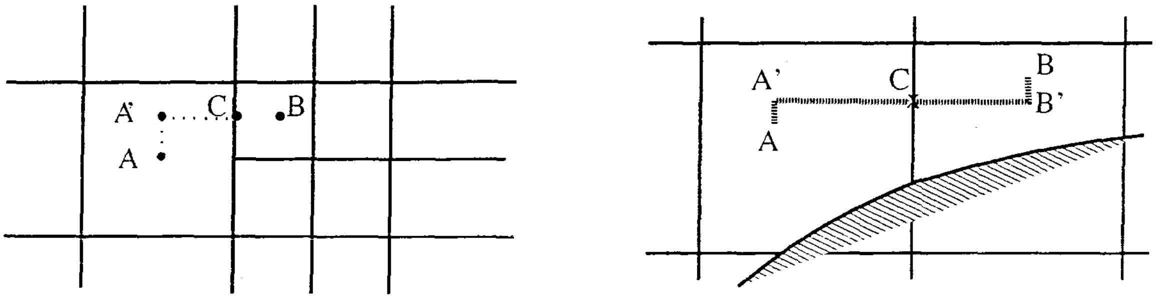
To give an example in two dimensions (see figure 9a), suppose we re-center the data in y at point A, to limit against the x face using point B on the other side. Let the face centroid values be denoted by a subscript m. This gives a recentered value at the virtual point A': $u_i^V = u_i + (y_m - y_i)\phi_y u_y$. The point at the interface is then

$$u_{i+1/2} = A' + h/2 \cdot \phi_x \cdot u_{x_i} \tag{32}$$

and ϕ_x is determined based only on the x gradient.

In reality this implicit system is too expensive to solve, and we approximate it instead using only a single iteration. This means the re-centering will use a possibly not yet limited component of the gradient, which is not robust. In practise for robustness we follow the limiter determination with a test for positivity. However, at most mesh refinement interfaces, only one side will need re-centering. By re-centering after the other faces where there is no mesh refinement have already been used for limiting, this use of partially-limited gradients has proven safe.

The cut cells in a Cartesian scheme may be extremely irregular, as illustrated in 9(b). For these cells, the re-centering is not a safe procedure. The gradient may not have been limited at all, so using it for re-centering in one direction to limit in another may produce unphysical results. For these more irregular cases we have developed the fully general procedure in the next subsection.



(a) (b)

Figure 9: (a) Illustration of re-centering on cut cells. (b) The data at a refinement interface is re-centered from A to A' before applying the limiter to prevent this.

4.2 Multi-Dimensional Limiting for Irregular Grids

In one dimension for $R > 0$ recall that the van Leer limiter

$$\phi_{VL} = \frac{4R}{(1+R)^2} = \frac{4}{2+\frac{1}{R}+R} = \frac{4}{2+\frac{D_-}{D_+}+\frac{D_+}{D_-}}. \tag{33}$$

On linear data, the ratio of the differences in (33) is 1, and the limiter is 1. When the solution is accelerating or decelerating, one of the ratios R or $1/R$ will be greater than 1 and the limiter will start to retard the gradient. To extend this to multiple dimensions our strategy is to replace the ratio of neighboring one-sided differences with the ratio of reconstructed least squares gradients in each coordinate direction. In addition, instead of the one-dimensional situation where there is one neighbor on the right and one on the left, we examine all face neighbors (reminiscent of the Benjamin limiter). The limiter ϕ in direction k for cell 0 is

$$\phi_k = \frac{4}{2+\frac{(\sum_{j\in nbors} d_j+\frac{1}{d_j})}{\#nbors}}, \qquad d_j = \frac{\nabla_k u_j}{\nabla_k u_0} \tag{34}$$

where k ranges over the coordinate directions x, y, z, and j is the index of a neighboring cell. Note that the use of the reconstructed gradient has expanded the stencil of the limiter. This formulation preserves independent limiting in each coordinate direction; ϕ_k is determined using the gradients in the $k_t h$ direction only. We can limit u_x without affecting u_y. For a linear function, the gradients in all cell are equal, the d_j's sum to $2 \cdot \#of\ neighbors$ (since the gradient ratio and its inverse are both counted), the denominator will sum to 4, and the limiter will be 1. Thus eq. (34) retains the property that linear solutions are not limited, and shuts off smoothly as $d \rightarrow 0$ or $d \rightarrow \infty$.

Unfortunately in one space dimension the generalization (34) becomes

$$\phi_j = \frac{4}{2 + \frac{\frac{D^c_{i+1}}{D^c_i} + \frac{D^c_i}{D^c_{i+1}} + \frac{D^c_{i-1}}{D^c_i} + \frac{D^c_i}{D^c_{i-1}}}{2}}. \tag{35}$$

and it is easy to show that this limiter is not always in the TVD region. One more modification is necessary, which we motivate by looking at the behavior of (35) in two specific cases. Since (35) involves a five-point stencil it is hard to compare graphically with the limiters in figure 2. Instead we look at two limiting cases: in figure 10(a) the solution extends linearly outside of u_{i-1}, u_i and u_{i+1}, and in figure 10(b) it is piecewise constant.

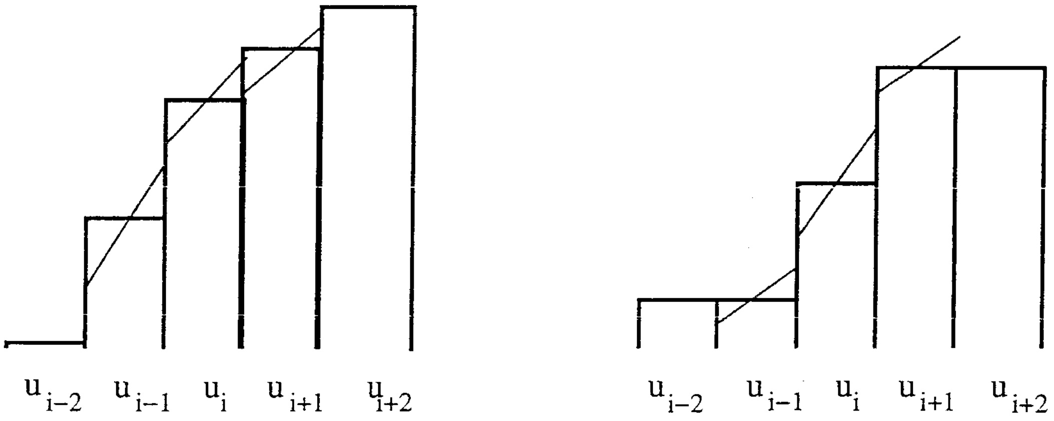


Figure 10: Examples using the 5 point stencil limiter in (34). The blue lines are the central difference slopes in each cell before limiting. In (a) the exact solution is linear outside the middle 3 cells, i.e. it is extended linearly to the right using u_i and u_{i+1} and to the left using u_i and u_{i-1}; in (b) the additional cells u_{i-2} and u_{i+2} are piecewise constant extensions. The usual van Leer limiter would not limit in case (b) for a value of u_i in the middle of the jump.

Let the jump $u_{i+1} - u_{i-1} = J$, and take u_i to be a fraction of the jump so that $u_i - u_{i-1} = f\,J$, $0 \leq f \leq 1$. For simplicity assume a uniform grid with mesh width $h = 1$. We then have for the forward, backward and central differences

$$\begin{aligned} D^+_i &= (1-f)J \\ D^-_i &= fJ \\ D^c_i &= J/2 \\ D^c_{i+1} &= (u_{i+2} - u_i)/2 = (D^+_{i+1} + D^-_{i+1})/2 = D^-_{i+1} = D^+_i \\ D^c_{i-1} &= D^-_i. \end{aligned}$$

For this case, we compute both the new limiter value and the standard van Leer limiter values to get

$$\phi_{new} = \frac{4}{3 + \frac{1}{4f(1-f)}} \tag{36}$$

$$\phi_{VL} = \frac{4}{2 + \frac{(1-f)^2+f^2}{f(1-f)}} \tag{37}$$

For $f = .5$, when u_i is exactly in the middle, both limiters are one, but for $0 < f < 1$, $\phi_{new} > \phi_{VL}$. In fact, a short calculation shows that there is a range of values for f for which (36) is outside the TVD region.

Comparing (36) and (37) suggests that we tweak the coefficients in (35), (the numbers 4,2, 1/2), to find a TVD alternative. For example several alternate possibilities in one dimension to the van Leer limiter are

$$\phi_{VL_1} = \frac{3}{1 + \frac{(\sum_{j\in nbors} d_j + \frac{1}{d_j})}{\#nbors}} \quad \text{and} \tag{38}$$

$$\phi_{VL_2} = \frac{4}{1 + \frac{\frac{3}{2}\cdot(\sum_{j\in nbors} d_j + \frac{1}{d_j})}{\#nbors}} \tag{39}$$

The formulation we recommend comes from looking at a second limiting case, when the solution is piecewise constant outside of the region of interest, shown in figure 10(b). Substituting for all the gradients and again comparing the new and usual van Leer limiter values in eq. (35) and (14) yields

$$\phi_{new} = \frac{4}{2 + \frac{(1+\frac{1}{f(1-f)})}{2}} \tag{40}$$

$$\phi_{VL} = \frac{4}{2 + \frac{1-f}{f} + \frac{f}{1-f}} \tag{41}$$

Again both limiters shut off when $f = 0$ or 1, when the intermediate value for u_i makes a sharp discontinuity. For $f = .5$, $\phi_{VL} = 1$, since it sees only a three point linear solution, but eq. (35) gives $\phi_{new} = 4/4.5$, so the steeper gradient in the middle is reduced. But again eq. (35) is not always in the TVD region. Looking further at (41) we rewrite

$$\phi_{VL} = \frac{4}{2 + \frac{1-f}{f} + \frac{f}{1-f}} = \frac{4}{2 + \frac{1}{f} - 1 + \frac{1}{1-f} - 1} = \frac{4}{\frac{1}{f} + \frac{1}{1-f}}$$

This suggests the final form we use in multi-dimensional calculations, which is

$$\phi_{VL_3} = \frac{2}{\frac{(\sum_{k\in nbors} d_k + \frac{1}{d_k})}{\#nbors}} \tag{42}$$

Interestingly, if (42) is returned to one dimension and d is replaced by R, it is the van Albada limiter. The fine-tuning of coefficients give a range of limiters lying between van Leer and van Albada in terms of dissipation. In figure 11, all the variations of the van Leer variation are plotted in their symmetric form.

All experiments in section 5 use eq. (42) for the general polyhedral cells, and take the sum over all face neighbors in all directions. A less dissipative multi-dimensional implementation could consider using only neighbors in the j^{th} direction instead of all neighbors when limiting the slope in the j^{th} direction. Alternatively, this same idea of looking at ratios of reconstructed gradients could be investigated for a different limiter, but we did not investigate this further.

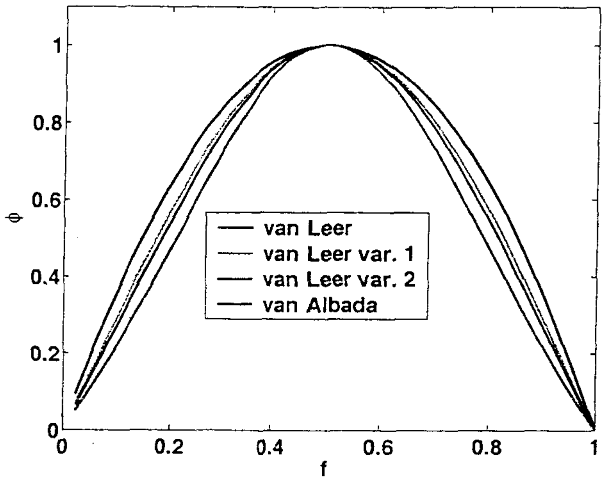


Figure 11: Variations on the van Leer slope limiter using the modified coefficients of eq. (38), (39), and eq. (42), which is the same as van Albada when d is replaced by R in one space dimension.

5 Computational Results

In this section we demonstrate the performance of the new limiters in several test cases. We solve the Euler equations for compressible inviscid flow using the flow solver from Cart3D [1]. This approach uses multi-level Cartesian meshes with embedded boundaries, so cells of all types of irregularities are encountered during the solution process. At the regular cells over most of flow domain, the limiter is applied in the usual way, using whichever limiter is specified. At mesh interfaces we use the re-centering eq. (34) with the specified limiter. The new multi-dimensional formulation of eq. (42) is used at the cut cells, regardless of which limiter is used elsewhere.

Since we expect all limiters to fire when there is a discontinuity in the solution, we instead examine their behavior on a smooth flow in three dimensions. We compute flow over an OneraM6 wing, with a free stream Mach number of .54 and angle of attack 3.06. This example doesn't need limiters to converge if started from a coarser solution using the full multi-grid algorithm. This makes the comparison between a scalar limiter and the new limiter formulation in figure 12 more evident. The first four grid levels in this full multigrid scheme do not use gradients or limiters, so the behavior of the residual is identical. The gradients and limiters are activated when the last level is reached. The new limiter and no limiter runs have essentially identical convergence behavior in figure 12. The scalar limiter by constrast seems to hang after only an order of magnitude in convergence. These experiments used a V cycle with one pre-sweep, and a five stage Runge-Kutta method where gradients were evaluated only at the first stage.

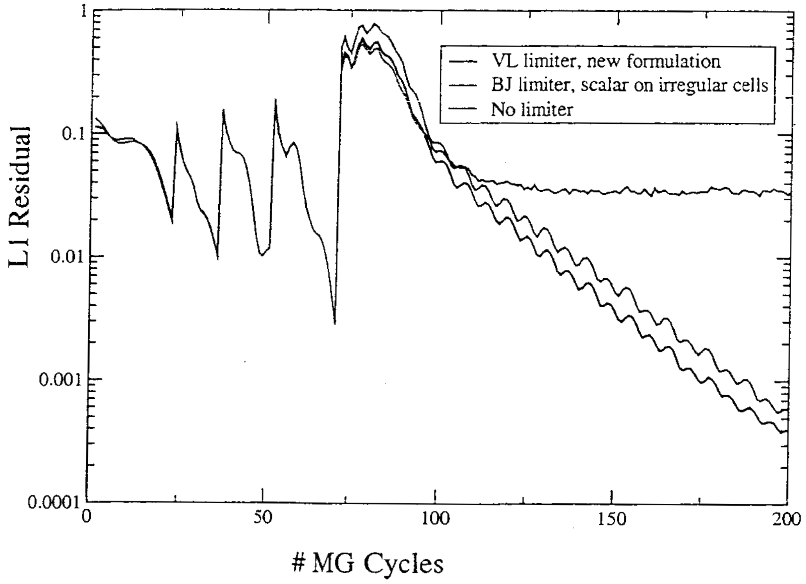


Figure 12: Convergence rates for subsonic OneraM6 wing example.

Figure 13 shows the same convergence histories using the new formulation with the five different limiters of figure 3. The respective limiters are applied over the flow domain and at mesh interfaces using eq. (32). At cut cells next to the wing only the new van Leer formulation eq. (42) is used. Not surprisingly, the more dissipative the limiter the better the convergence. Barth-Jespersen, as before, has the most problems with convergence. Note however that this formulation is much improved over the version of figure 12, which is scalar at the cut cells and not linearity preserving at the mesh interfaces. It makes a surprisingly large difference, even though those two categories of cells are only a small fraction of the mesh.

Although it is hard to say what the "exact" answer is, we expect the unlimited results to be the best on this mesh, and the answers to deteriorate with increasing dissipation. More evidence in this direction is seen by looking at the computed drag, which should converge to zero for this subsonic case. Table 1 gives the drag in order of increasing dissipation. All the limiters should converge as the grid is refined, but for a given grid, the less dissipative limiter gives the better drag count. This was computed on a rather coarse grid of 202K cells, to make the results more obvious.

The last example shows the performance of the new limiter on a problem with extremely complicated geometry and physics. The geometry is rather coarsely resolved, with only 1.8M cells in this calculation. The freestream Mach number is 18.5, with angle of attack 15 degrees. The density and pressure in the OMS pods and engine bells gets as low as 10^{-7}. The pressure on the orbiter's nose is nearly 450 times that in the freestream, a range of nearly eight orders of magnitude. Exactly the same limiter settings with no special tweaking are used for this case, despite the deep concavities, shown in figure 14(a). Figure 15 shows the convergence history of the residual and the forces for this case. This example used a

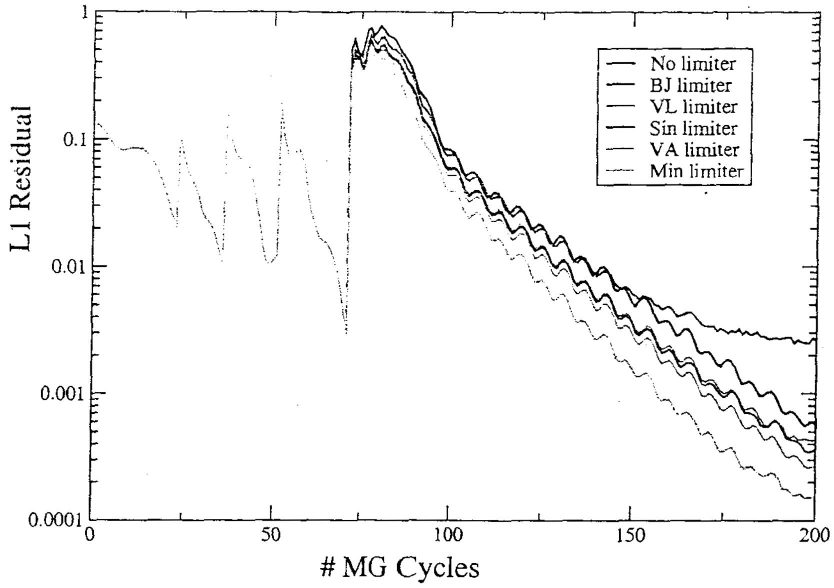


Figure 13: Convergence rates for subsonic OneraM6 wing example.

Table 1: Computed drag for different limiters, in order of increasing dissipation.

Limiter Type	Computed Drag
No limiter	1.568e-03
BJ limiter	2.230e-03
VL limiter	2.797e-03
Sin limiter	2.949e-03
VA limiter	3.335e-03
Min limiter	4.669e-03

3 levels of a multigrid W cycle, with both a pre-sweep and post-sweep. Gradients were evaluated at every stage of the Runge-Kutta scheme.

6 Conclusions

We have reviewed and analyzed common forms of gradient limiters, highlighting their shortcomings for practical application. This examination revealed the failure of traditional implementations to preserve linear solutions in the presence of mesh stretching or other irregularities, and a lack of a clear extension to multiple dimensions. From this analysis a new formulation was developed which preserves linearity in the general multi-dimensional case while maintaining the smoothness properties of the original limiter. The performance of the new limiter was demonstrated in three dimensional applications using a Cartesian embedded-boundary method. These examples illustrate the improved convergence and accuracy properties relative to traditional formulations, as well as the robustness of the new

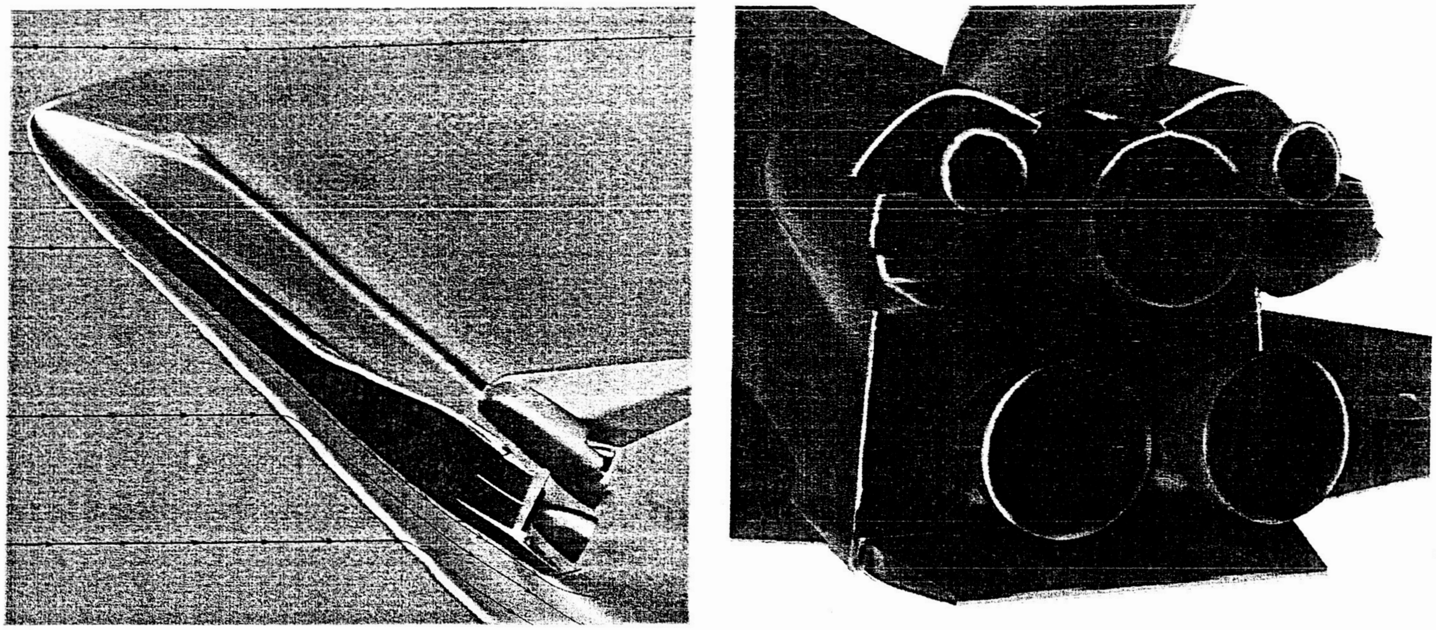


Figure 14: Orbiter calculation using 1.8M cell mesh, Mach 18.5 at 15 degrees angle of attack. The contours show the log of pressure. The left figure also shows some streamlines of the flow in black. Both the complex geometry and complicated physics of the flow field are apparent.

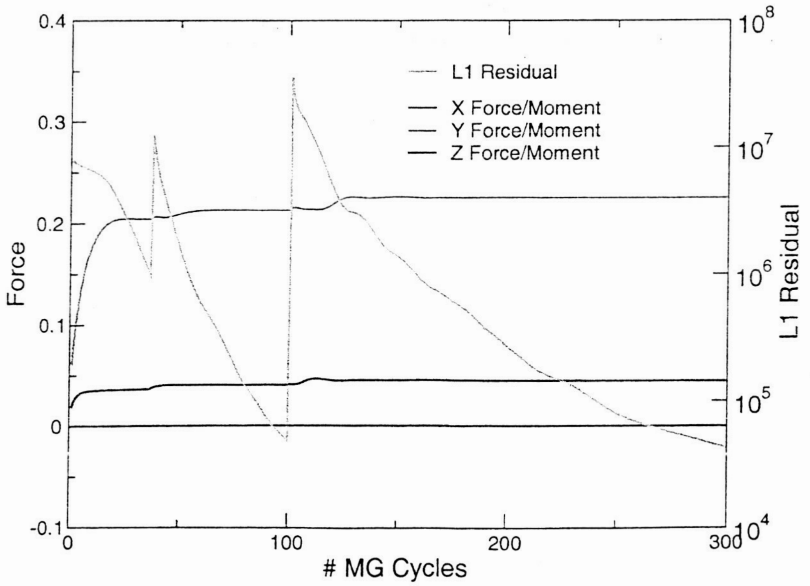


Figure 15: Convergence rate of L1 residual for orbiter calculation. Also shown are the computed forces at each iteration.

form.

While this work has focussed on linearity preserving limiters, a rich area for future work is limiting for higher order methods. This is especially true in multiple spatial dimensions where the savings from higher order schemes are particularly attractive.

7 Acknowledgements

We thank Jonathan Goodman for insightful discussions. Marsha Berger was supported by AFOSR grant F19620-00-0099 and by DOE grants DE-FG02-00ER25053 and DE-FC02-01ER25472.

References

[1] M. Aftosmis, M. Berger, and G. Adomavicius. A parallel multilevel method for adaptively refined Cartesian grids with embedded boundaries. *AIAA-2000-0808*, January 2000.

[2] M. Aftosmis, D. Gaitonde, and T. Sean Tavares. On the accuracy, stability and monotonicity of various reconstruction algorithms for unstructured meshes. *AIAA-94-0415*, January 1994.

[3] Tim Barth. Numerical methods and error estimation for conservation laws on structured and unstructured meshes. In *Von Karman Institute Computational Fluid Dynamics Lectures Notes*, 2003.

[4] P. Colella and P. Woodward. The piecewise parabolic method (PPM) for gas-dynamical simulations. *J. Comp. Phys.*, 54(1):174–201, 1984.

[5] J.B. Goodman and R.J. LeVeque. A geometric approach to high resolution TVD schemes. *SIAM J. Num. Anal.*, 25, 1988.

[6] A. Harten. High resolution schemes for hyperbolic conservation laws. *J. Comp. Phys.*, 49:357–393, 1983.

[7] M.E. Hubbard. Multidimensional slope limiters for MUSCL-type finite volume schemes on unstructured grids. *J. Comp. Phys.*, 155:54–74, 1999.

[8] Carl Ollivier-Gooch. Quasi-ENO schemes for unstructured meshes based on unlimited data-dependent least-squares reconstruction. *J. Comp. Phys.*, 133:6–17, 1997.

[9] W. J. Rider and D. B. Kothe. Constrained minimization for monotonic reconstruction. *AIAA-97-2036*, 1997.

[10] Jeff Saltzman. Monotonic difference schemes for the linear advection equation in two and three dimensions. Technical Report 87-2479, Los Alamos National Laboratory, 1987.

[11] S.P. Spekreijse. Multigrid discretization of monotone second-order discretizations of hyperbolic conservation laws. *Math. Comp.*, 49, 1987.

[12] P.K. Sweby. High resolution schemes using flux limiters for hyperbolic conservation laws. *SIAM J. Num. Anal.*, 21, 1984.

[13] V. Venkatakrishnan. Convergence to steady state solutions of the Euler equations on unstructured grids with limiters. *J. Comp. Phys.*, 118:120–130, 1995.

[14] S. T. Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *J. Comp. Phys.*, 31:335–362, 1979.

This work is an extended abstract to be submitted to the *43nd Aerospace sciences meeting and exhibit* session on applied CFD. The conference will be in Reno NV Jan 6-11, 2005.

This paper examines the behavior of flux and slope limiters on non-uniform grids in multiple dimensions. We note that on non-uniform grids the standard scalar formulation in practical use today sacrifices *k*-exacctness even for linear solutions, impacting both accuracy and convergence. We rewrite some well-known limiters in a new way to highlight their underlying symmetry, and use this to examine both traditional and new limiter formulations. We develop a new directional limiter in multiple dimensions, and show results demonstrating improved accuracy and convergence using a combination of model problems and complex three-dimensional examples.